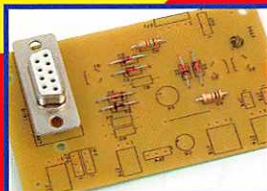
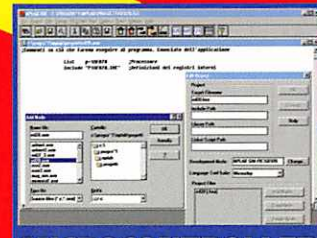
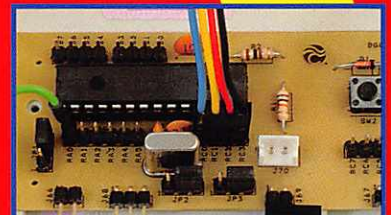
...e costruisci il tuo LABORATORIO DIGITALE

6,90 €

28

HARDWARE

DIGITALE DI BASE

MICROCONTROLLER

DIGITALE AVANZATO

TOTALMENTE PROGRAMMABILE!!!

9 771824 363008 41028

Peruzzo & C.

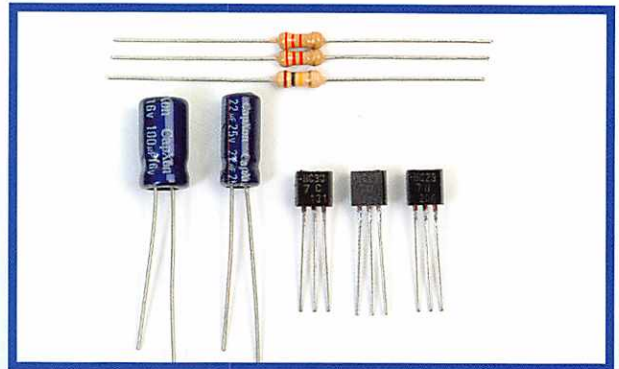
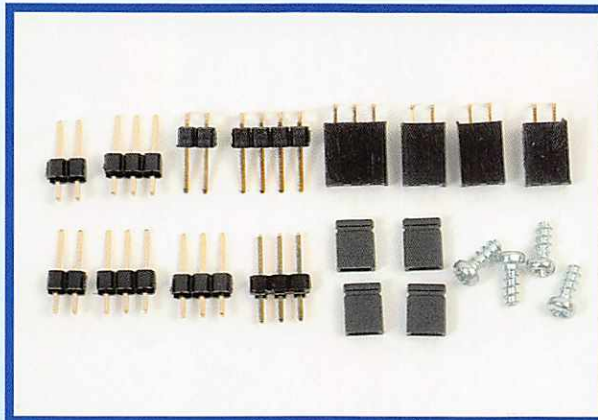
Direttore responsabile:
ALBERTO PERUZZO
Direttore Grandi Opere:
GIORGIO VERCELLINI
Consulenza tecnica e traduzioni:
CONSULCOMP S.n.c.
Pianificazione tecnica
LEONARDO PITTON

Direzione, Redazione, Amministrazione: viale Ercole Marelli 165, Tel. 02/242021, 20099 Sesto San Giovanni (Mi). Pubblicazione settimanale. Registrazione del Tribunale di Monza n. 1738 del 26/05/2004. Spedizione in abbonamento postale gr. II/70; autorizzazione delle Poste di Milano n. 163464 del 13/2/1963. Stampa: Grafiche Porpora s.r.l., Cernusco S/N (MI). Distribuzione SO.DI.P. S.p.A., Cinisello Balsamo (MI).

"ELETTRONICA DIGITALE"
si compone di
70 fascicoli settimanali
da suddividere
in 2 raccoglitori.

RICHIESTA DI NUMERI ARRETRATI. Per ulteriori informazioni, telefonare dal lunedì al venerdì ore 9.30-12.30 all'ufficio arretrati tel. 02/242021. Se vi mancano dei fascicoli o dei raccoglitori per completare l'opera, e non li trovate presso il vostro edicolante, potrete riceverli a domicilio rivolgendovi direttamente alla casa editrice. Basterà compilare e spedire un bollettino di conto corrente postale a PERUZZO & C. s.r.l., Ufficio Arretrati, viale Marelli 165, 20099 Sesto San Giovanni (MI). Il nostro numero di c/c postale è 42980201. L'importo da versare sarà pari al prezzo dei fascicoli o dei raccoglitori richiesti, più le spese di spedizione € 3,10 per pacco. Qualora il numero dei fascicoli o dei raccoglitori sia tale da superare il prezzo globale di € 25,82 e non superiore a € 51,65, l'invio avverrà per pacco assicurato e le spese di spedizione ammonteranno a € 6,20. La spesa sarà di € 9,81 da € 51,65 a € 103,29; di € 12,39 da € 103,29 a € 154,94; di € 14,98 da € 154,94 a € 206,58; di € 16,53 da € 206,58 in su. Attenzione: ai fascicoli arretrati, trascorse dodici settimane dalla loro distribuzione in edicola, viene applicato un sovrapprezzo di € 0,52, che andrà pertanto aggiunto all'importo da pagare. Non vengono effettuate spedizioni contrassegno. Gli arretrati di fascicoli e raccoglitori saranno disponibili per un anno dal completamento dell'opera. IMPORTANTE: è assolutamente necessario specificare sul bollettino di c/c postale, nello spazio riservato alla causale del versamento, il titolo dell'opera nonché il numero dei fascicoli e dei raccoglitori che volete ricevere.

IN REGALO in questo fascicolo

- 1 Transistor BC307
- 2 Transistor BC237
- 1 Condensatore elettrolitico da 100 µF
- 1 Condensatore elettrolitico da 22 µF
- 2 Resistenze da 2K2 5% 1/4 W
- 1 Resistenza da 100K 5% 1/4 W

IN REGALO nel prossimo fascicolo

- 3 Connettori femmina da c.s. a due vie a 90°
- 1 Connettore femmina da c.s. a tre vie a 90°
- 1 Connettore maschio da c.s. a due vie a 90°
- 1 Connettore maschio da c.s. a quattro vie a 90°
- 4 Connettori maschio da c.s. a tre vie diritti
- 2 Connettori maschio da c.s. a due vie diritti
- 4 Ponticelli isolati
- 4 Viti

COME RACCOGLIERE E SUDDIVIDERE L'OPERA NELLE 4 SEZIONI

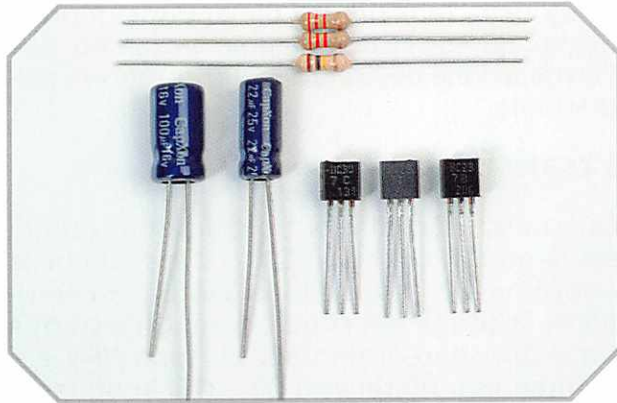
L'Opera è composta da 4 sezioni identificabili dalle fasce colorate, come indicato sotto. Le schede di ciascun fascicolo andranno suddivise nelle sezioni indicate e raccolte nell'apposito raccoglitore, che troverai presto in edicola. Per il momento, ti consigliamo di suddividere le sezioni in altrettante cartellette, in attesa di poterle collocare nel raccoglitore. A prima vista, alcuni numeri di pagina ti potranno sembrare ripetuti o sbagliati. Non è così: ciascuno fa parte di sezioni differenti e rispecchia l'ordine secondo cui raccogliere le schede. **Per eventuali domande di tipo tecnico scrivere al seguente indirizzo e-mail:** elettronicadigitale@microrobots.it

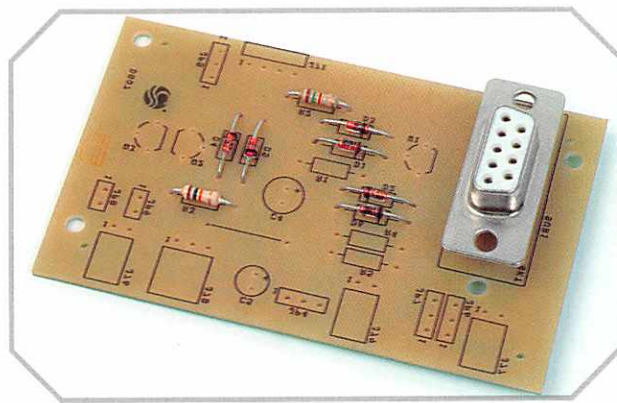
Hardware Montaggio e prove del laboratorio

Digitale di base Esercizi con i circuiti digitali

Digitale avanzato Esercizi con i circuiti sequenziali

Microcontroller Esercizi con i microcontroller

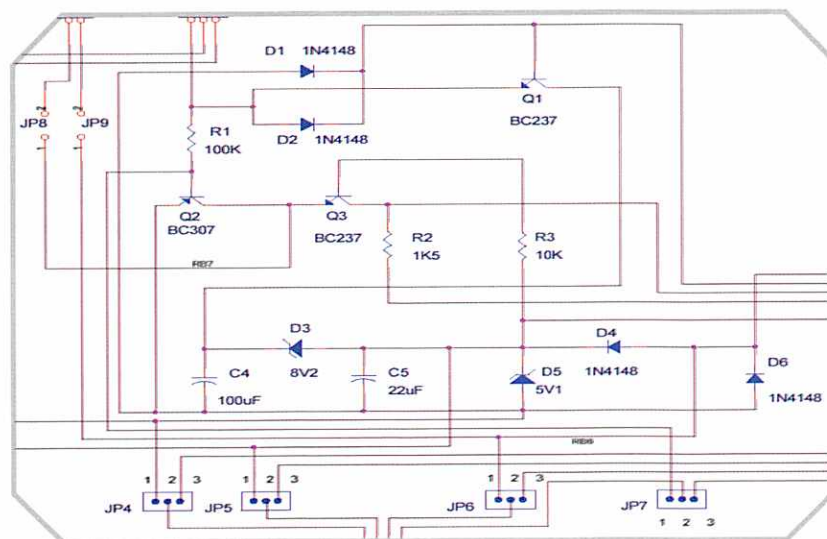
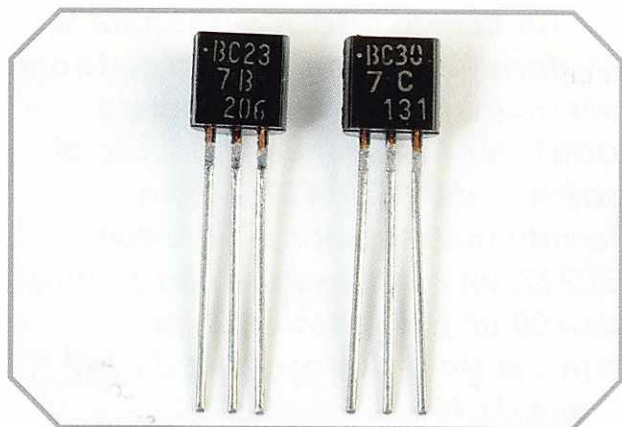
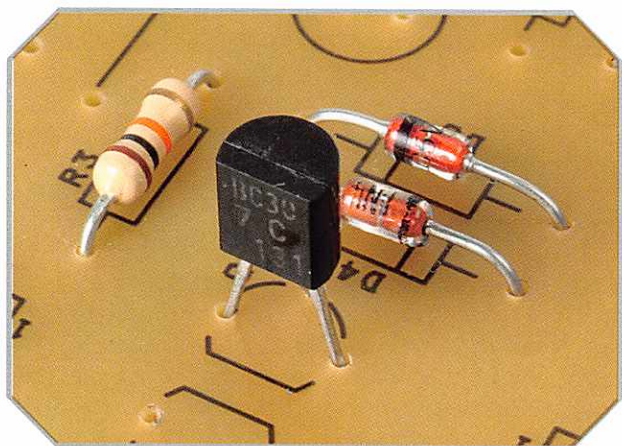
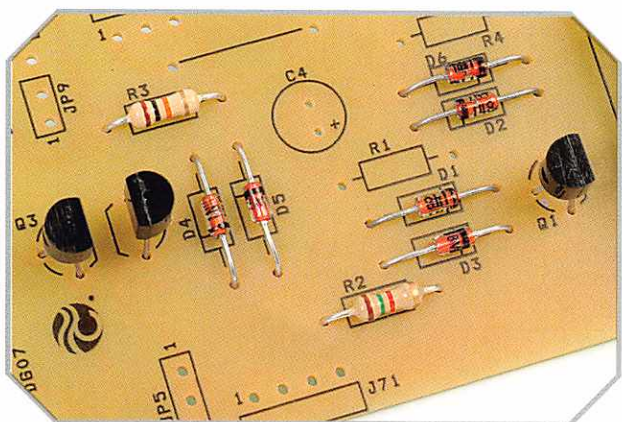
Circuito di scrittura (II)

Componenti elettronici forniti con questo fascicolo.

Con questo fascicolo continua la fornitura di componenti elettronici per montare il circuito stampato DG07 che corrisponde al circuito di scrittura del PIC 16F870. Viene fornito un transistor BC307, due BC237, un condensatore elettrolitico da 100 µF e un altro da 22 µF e infine tre resistenze, due da 2K2 e una da 100 K.

Il circuito stampato DG07 deve essere a questo livello di montaggio.

Tensioni di scrittura

Quasi tutti i componenti elettrolitici di questo circuito stampato sono utilizzati per costruire il circuito che genera le tensioni di scrittura, prendendo l'energia dalla porta seriale stessa del computer a cui ci collegheremo. Gli unici componenti che non fanno parte di questo circuito sono le resistenze R4 e R5 da 2K2 che sono di pull up.

Montaggio

Continuiamo il montaggio partendo dai componenti che dovrebbero essere già montati, non per seguire un ordine di montaggio che in questo caso non è importante, ma perché installando i componenti ci sono meno probabilità di perderli, situazione che potrebbe verificarsi se non si collocano a tempo debito, a

Schema elettrico del circuito DG07. Zona ampliata del circuito che genera le tensioni di scrittura.

Si utilizzano due tipi di transistor BC307 e BC237.

Il transistor BC307 si monta sulla posizione Q2.

I transistor del tipo BC237 si montano su Q1 e Q3.

causa della loro ridotta dimensione. Riprendiamo quindi l'assemblaggio iniziando dall'installazione dei tre transistor contenuti dalla scheda.

I transistor

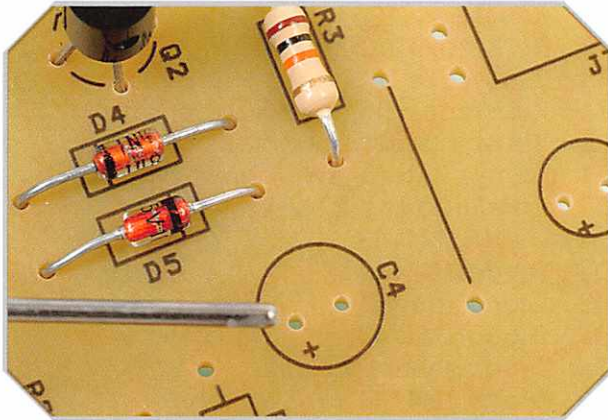
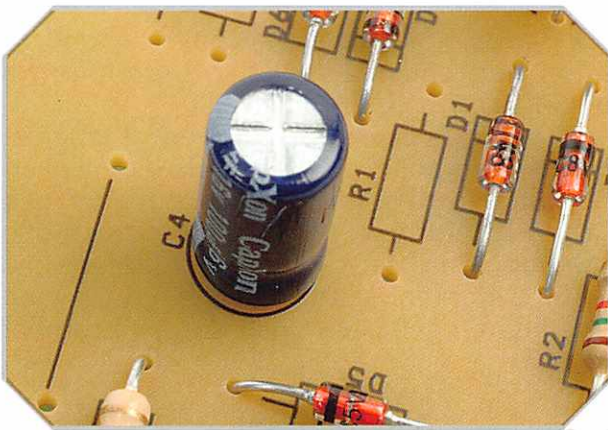
Questa scheda utilizza tre transistor, uno di essi è un PNP, di tipo BC307, si installa nella posizione Q2, seguendo la serigrafia che riporta la forma del contenitore. Gli altri due transistor sono di tipo BC237, sono NPN e si montano sui riferimenti Q1 e Q3. Bisogna fare molta attenzione a non scambiare il transistor Q2 con qualcuno degli altri, altrimenti il circuito non funzionerebbe e potrebbe causare la distruzione del PIC.

La ridotta dimensione di questi componenti, permette un montaggio non a filo della scheda, ma sollevati di 4 o 5 millimetri.

La saldatura deve essere eseguita applicando la punta del saldatore per un tempo sufficiente a far sciogliere lo stagno e ottenere così un buon risultato. È piuttosto frequente trovare delle saldature fredde su questo tipo di componenti, perché è stato applicato il saldatore per un tempo insufficiente a far scogliere lo stagno e a portare il terminale del componente a un livello di temperatura idoneo per realizzare una saldatura corretta. A questo problema si può ovviare in parte, utilizzando stagno per applicazioni di elettronica, infatti lo stagno utilizzato in applicazioni di carpenteria leggera anche se di eccellente qualità, ha una temperatura di fusione molto più alta e non è progettato per questo utilizzo.

D'altra parte i componenti elettronici dotati di terminali da saldare sopportano bene la saldatura, sempre che sia realizzata in modo adeguato. Per preservare il componente dalle temperature della saldatura è utile anche prendere il terminale con una pinza metallica: dato che questa conduce il calore, la temperatura si abbassa ulteriormente.

Condensatori

I due condensatori utilizzati su questa scheda sono di tipo elettrolitico, cioè hanno polarità, indicata normalmente sul contenitore del condensatore stesso, dove viene indicato il terminale negativo. Il terminale positivo solitamente è più lungo, anche se non sempre è così.

Il terminale positivo dei condensatori elettrolitici è indicato sulla serigrafia della scheda.

Il condensatore C4 è elettrolitico, da 100 μF.

Durante il montaggio di questi componenti bisogna fare attenzione a inserire il terminale positivo nel foro segnato con il segno +. Su questa scheda si utilizza un condensatore da 100 μF con sigla C4 e un altro da 22 μF con sigla C5.

Resistenze

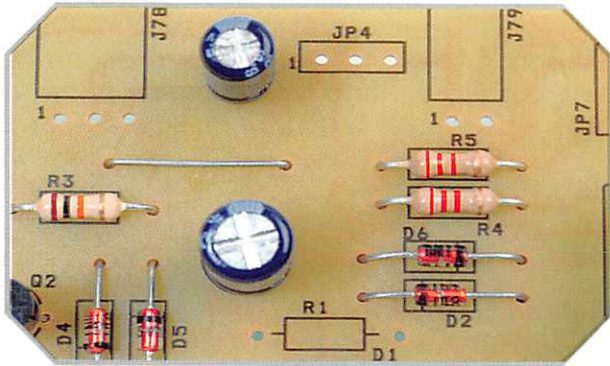
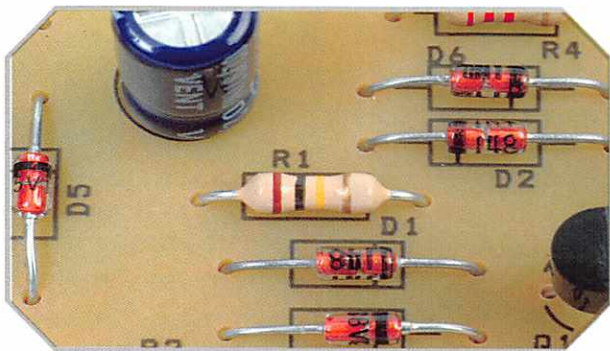
Rimangono da installare tre resistenze, R4 e R5 da 2K2 (rosso, rosso, rosso) e R1 da 100 K (marrone, nero, giallo). Si montano come già spiegato nei fascicoli precedenti, saldando i terminali e tagliando successivamente la parte in eccesso degli stessi.

Il ponticello

Il pezzo che avanza dei terminali delle resistenze, o semplicemente un pezzo di filo spelato, può essere utilizzato per realizzare un ponticello sulla superficie della scheda, che si può vedere vicino alla resistenza R3. È sufficiente inserire il filo dal lato dei componenti e saldare gli estremi come se si trattasse di un normale componente.

Revisione

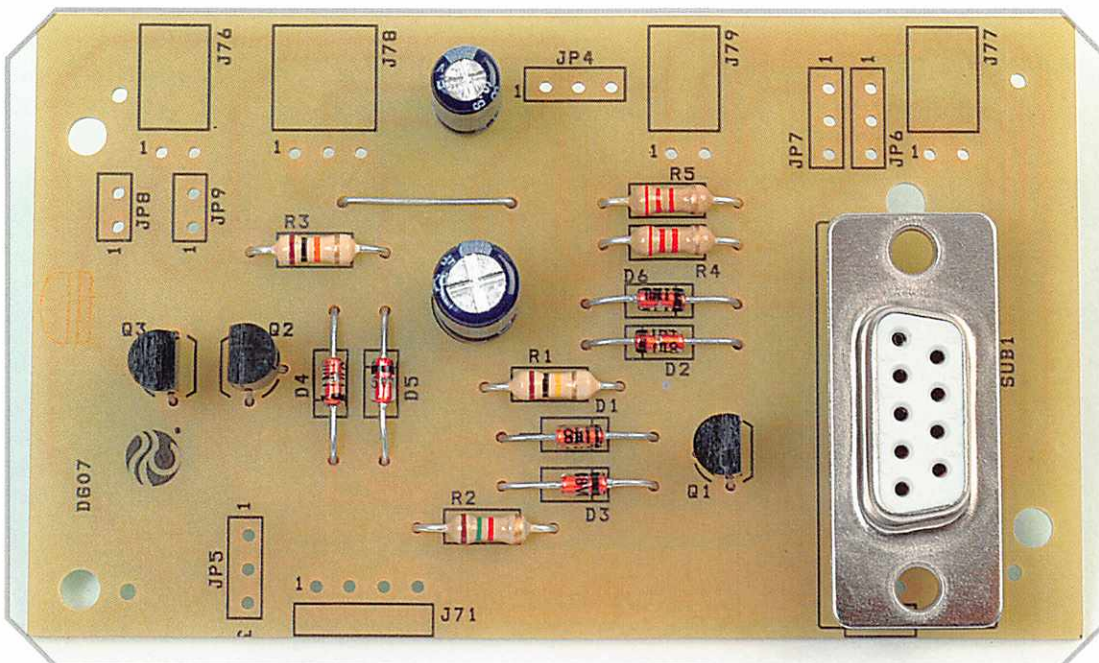
Il montaggio di questa scheda è vitale per la scrittura del PIC, pertanto bisogna verificare che ogni componente installato sia quello giusto. In primo luogo verificheremo la posizione di ogni componente seguendo il suo riferimento. Dobbiamo trovare i seguenti componenti:

Il condensatore C5 è elettrolitico, da 22 μF.

Resistenze da 2K2 nelle posizioni R4 e R5. Si può anche osservare il ponticello realizzato sulla scheda.

La resistenza da 100 K va nella posizione R1.

R1 Resistenza 100 K (marrone, nero, giallo)
R2 Resistenza 1K5 (marrone, verde, rosso)
R3 Resistenza 10 K (marrone, nero, arancio)
R4 Resistenza 2K2 (rosso, rosso, rosso)
R5 Resistenza 2K2 (rosso, rosso, rosso)
C4 Condensatore 100 µF elettrolitico
C5 Condensatore 22 µF elettrolitico
D1 Diodo 1N4148
D2 Diodo 1N4148
D3 Diodo zener 8V2
D4 Diodo 1N4148
D5 Diodo zener 5V1
D6 Diodo 1N4148
Q1 Transistor NPN BC237
Q2 Transistor PNP BC307
Q3 Transistor NPN BC237

Il connettore a 9 pin deve già essere montato.

È necessario verificare anche che i condensatori elettrolitici e i diodi siano inseriti con la polarità corretta.

Girando la scheda potremo vedere le saldature dei componenti, verificando di non averne dimenticata nessuna e che tutte siano ben fatte. Per effettuare questo controllo in modo comodo ed efficace, conviene utilizzare una lente d'ingrandimento, le saldature devono circondare bene i terminali di tutti i componenti.

Ecco lo stato in cui si deve trovare la scheda in attesa di ulteriori componenti.

Temporizzatore di ingresso

Il circuito oggetto di questa prova è molto utilizzato, sia per forzare un livello iniziale in un circuito, che per ottenere ritardi nei circuiti d'allarme.

Il circuito

Osserviamo lo schema generale dell'esperimento, l'elemento base dello stesso è la porta U1A del circuito integrato 4001. Sul suo ingresso vediamo collegato un condensatore C1, una resistenza R5 e un commutatore A. Questa porta NOR ha i due ingressi uniti fra loro pertanto funziona come una porta invertente, quando sul suo ingresso c'è un livello 0, sulla sua uscita c'è un livello 1.

Anche l'altra porta del 4001 utilizzata nel circuito, la U1B, è impiegata come porta invertente; quando sul suo ingresso c'è un 1 sulla sua uscita c'è uno 0, e in questa circostanza il transistor Q1 è in interdizione, non conduce e i quattro LED rimangono spenti.

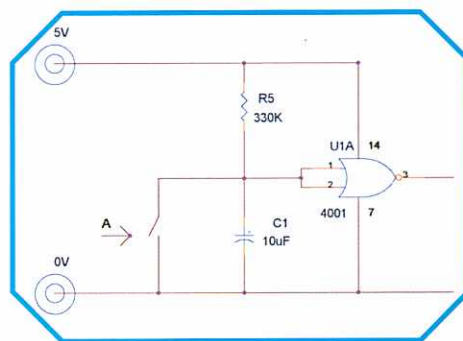
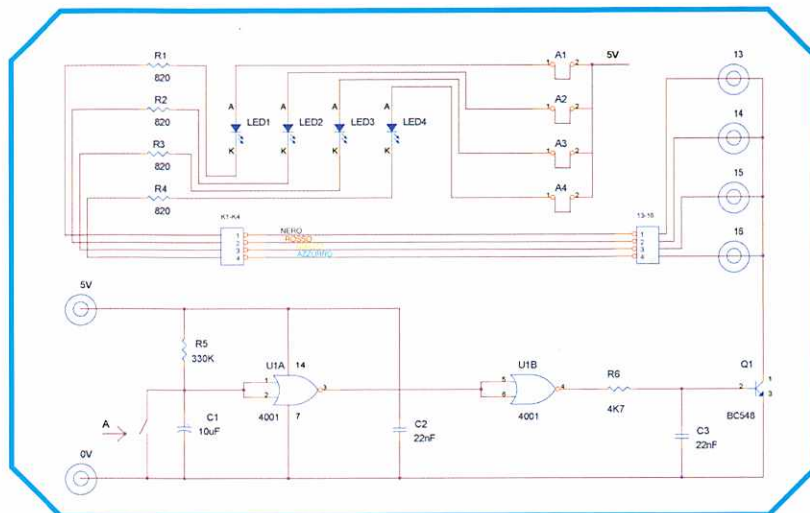
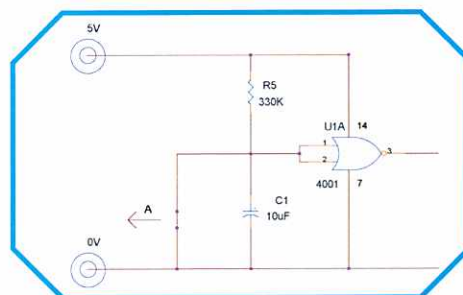
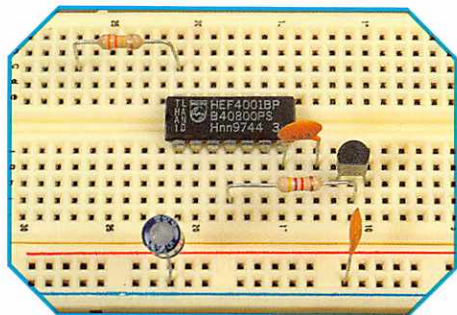
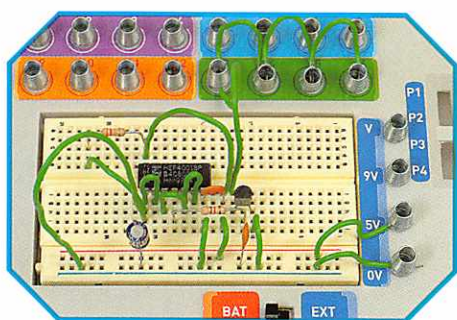
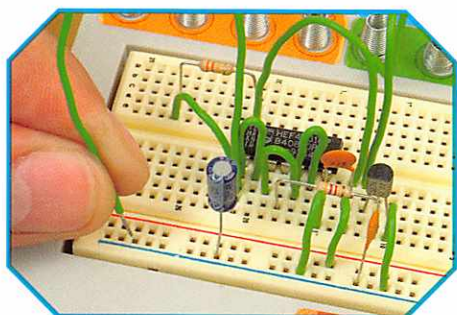
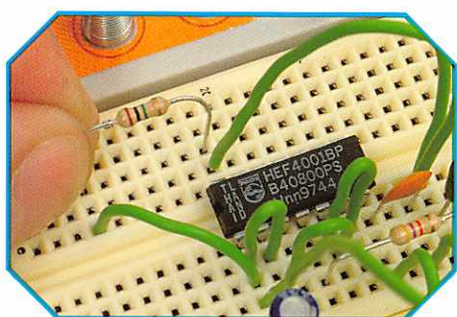
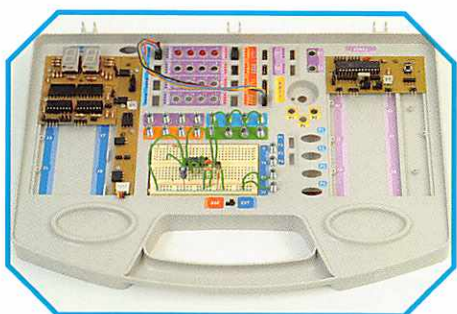
Funzionamento

Il funzionamento del circuito dipende dall'applicazione che determina le condizioni iniziali dello stesso.

Supponiamo che il commutatore A sia aperto, in questo caso al momento di collegare l'alimentazione, il condensatore è scarico e dato che la sua tensione non varia in modo istantaneo, ma aumenta in modo proporzionale alla carica che avviene tramite la resistenza R5, quando il livello di tensione del condensatore è sufficiente per fare in modo che l'ingresso della porta U1A lo interpreti come un 1, l'uscita di questa porta passa a livello basso, e quindi l'uscita dell'altra porta, la U1B passa a livello alto e si produce l'illuminazione dei LED. Il condensatore C3 si utilizza come filtro per i disturbi a frequenze elevate e produce un piccolo ritardo sul transistor, praticamente trascurabile.

Quando si chiude A si scarica rapidamente il condensatore e i LED si spengono.

Supponiamo ora di chiudere A prima di collegare l'alimentazione, in questo caso i LED non si illuminano. Dato che in questa situazione se apriamo il collegamento A inizia la carica del condensatore, i LED rimarranno spenti fino a quando il livello di tensione della carica del condensatore sarà interpretato come un 1 dall'ingresso della porta. Possiamo considerare due casi: se si lascia il collegamento aperto, il con-

Aprendo A si permette la carica del condensatore.

Con A chiuso si assicura uno zero all'ingresso del circuito.

Circuito temporizzatore di ingresso.

Componenti sulla scheda Bread Board.

Scheda Bread Board con tutti i collegamenti dell'esperimento.

Aprendo A inizia la temporizzazione del circuito.

Incrementando R5 aumenta il tempo di ingresso.

Laboratorio con l'esperimento.

densatore completerà la sua carica e i LED si illumineranno; potrebbe succedere però che si chiuda nuovamente il contatto A prima che il livello di tensione della carica del condensatore sia un 1, in questo caso i LED non si illumineranno.

Ipotizziamo di associare questo interruttore alla porta di accesso di un'abitazione, in modo che aprendo la porta si apra l'interruttore e chiudendola si chiuda. In questo esempio collegheremo il circuito, quindi se la porta è chiusa i LED rimangono spenti, se la porta è aperta la dovremo chiudere prima che si completi la carica del condensatore e si illuminino i LED. Inoltre la porta dovrà rimanere chiusa, perché se rimanesse aperta per più tempo di quanto programmato, i LED si illuminerebbero, indicandoci che abbiamo dimenticato di chiudere la porta.

Montaggio

Il montaggio di questo esperimento utilizza pochi componenti. È sufficiente collocarli bene ed eseguire il cablaggio in modo corretto, rispettando le polarità dei condensatori elettrolitici, la posizione del circuito integrato 4001 e quella del transistor BC548. È necessario collegare un cavetto a quattro fili tra i terminali corrispondenti ai catodi dei LED da 1 a 4 e ai terminali da 13 a 16 per ottenere sulle molle dal 13 al 16 i collegamenti dei catodi di questi LED.

Alimentazione

Dopo aver verificato il montaggio si collega l'alimentazione, il negativo di questa è 0 V e il positivo 5 V. Le pile del primo portabatterie, quello posizionato sotto la zona 1, sono sufficienti. Il commutatore di alimentazione deve essere nella posizione BAT; inoltre bisogna collocare i ponticelli tra gli anodi dei LED, da 1 a 4, e i terminali + che sono al loro fianco. Il condensatore C2 è un filtro di alimentazione.

LISTA DEI COMPONENTI

U2	Circuito integrato 4001
Q1	Transistor BC548 o BC547
R5	Resistenza 330 K (arancio, arancio, giallo)
R6	Resistenza 4K7 (giallo, viola, rosso)
C1	Condensatore 10 µF elettrolitico
C2, C3	Condensatore 22 nF

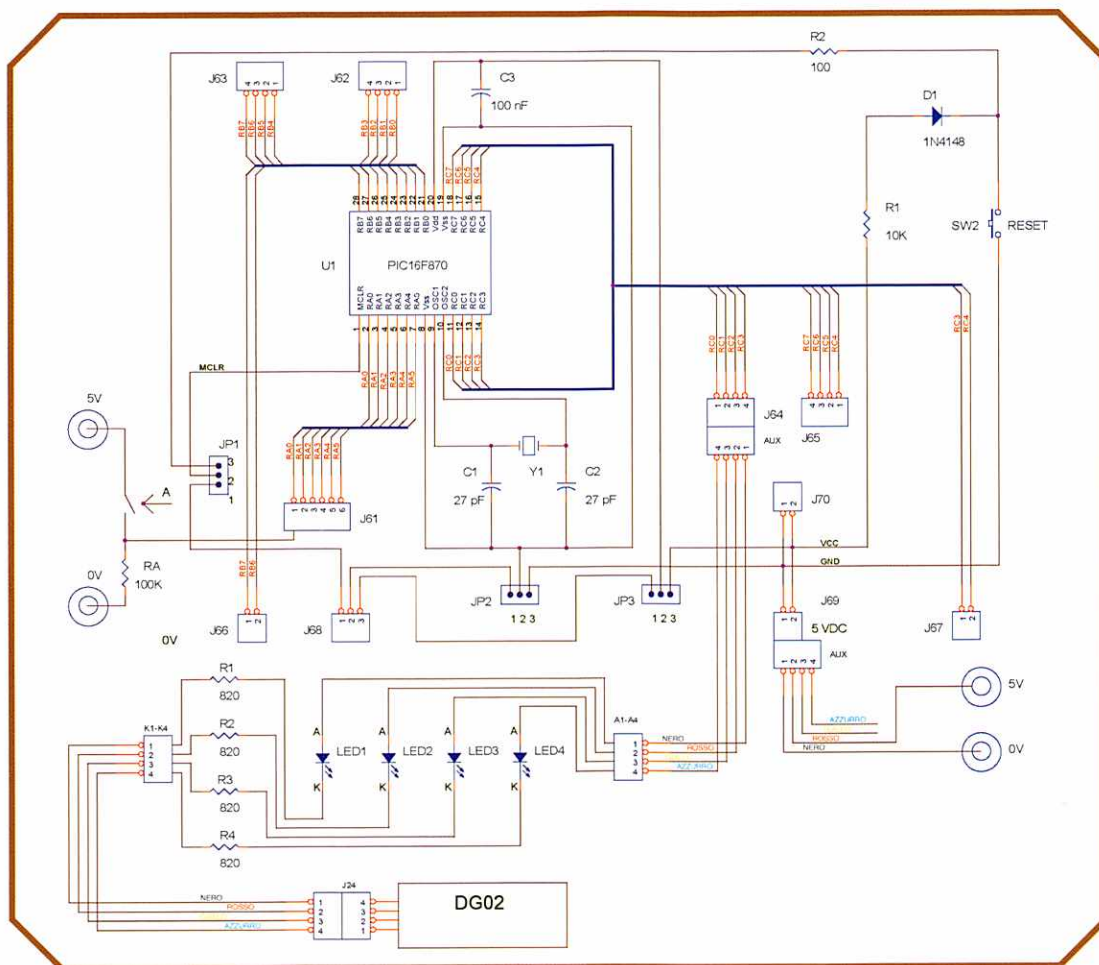
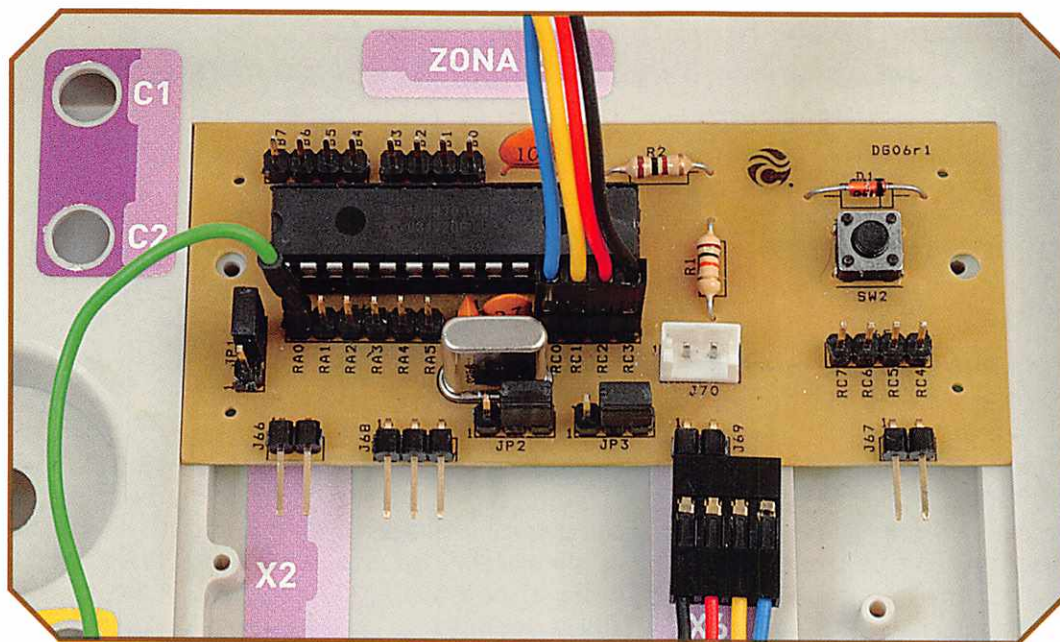
Secondo esperimento con il PIC16F870

In questa seconda prova che corrisponde al primo progetto, sappiamo già come è stato realizzato il programma. Si utilizza la stessa parte che il programma utilizzava nel primo esperimento, e le stesse uscite, terminali da RC3 a RC0 della porta C, per controllare l'accensione dei LED con cui si rappresentava il numero in binario.

Abbiamo realizzato un montaggio che risolve il nostro progetto, ma è possibile migliorarlo?

Come si può verificare questo progetto è stato risolto in modo molto semplice e relativamente economico, ma con uno scarso effetto visivo. Utilizzando un altro tipo di periferica possiamo migliorare la presentazione finale del progetto, anche se verranno penalizzati sia il costo finale che il tempo impiegato.

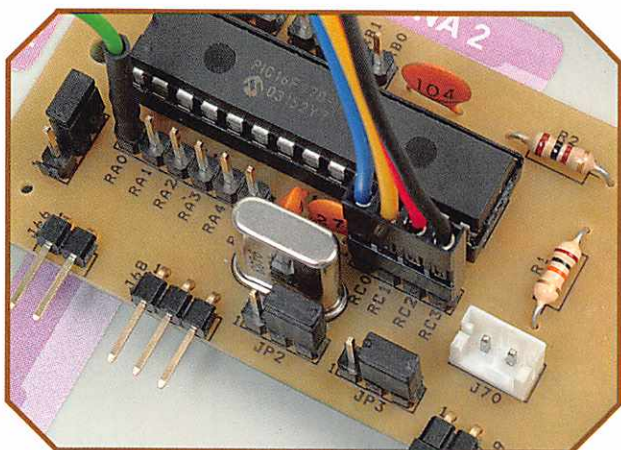
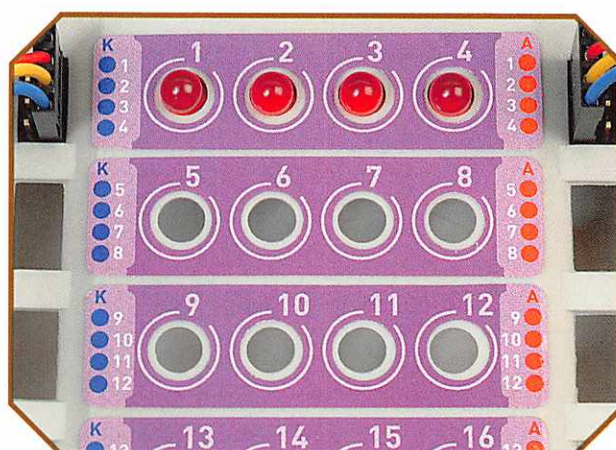
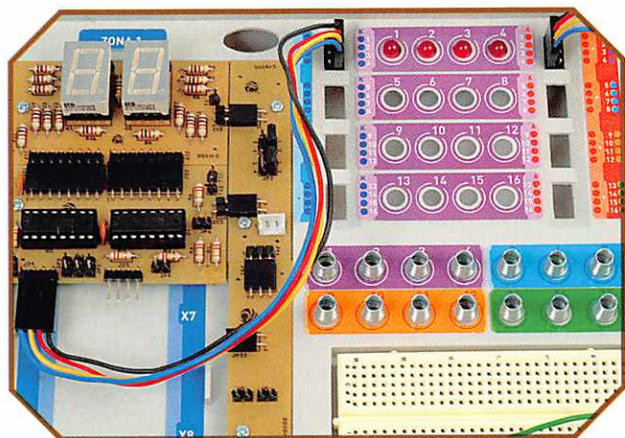
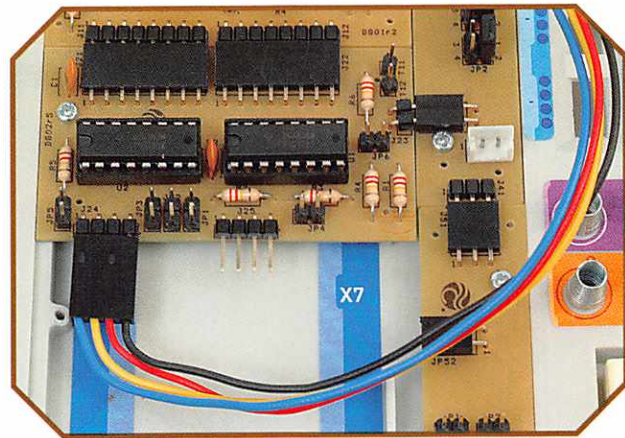
Utilizzo di display a 7 segmenti

La soluzione di rappresentare i numeri dallo 0 al 9 in codice binario è corretta ma poco pratica, quindi utilizzeremo questa stessa uscita applicata all'ingresso di uno dei driver 4511 della scheda DG02, per ottenere la visualizzazione del numero direttamente sul display a 7 segmenti.

Schema elettrico corrispondente all'esperimento.

Circuito stampato DG06, con i ponticelli in posizione normale.

Il vantaggio di utilizzare il driver è che dal punto di vista del funzionamento del PIC è esattamente uguale al primo esperimento.

Il lavoro con i driver

Una soluzione per vedere i numeri sul display a 7 segmenti è utilizzare i driver. L'integrato 4511 realizza una conversione interna di un numero in sistema binario al corrispondente numero in codice a 7 segmenti. In questo modo, se noi introduciamo sull'ingresso del driver il numero che vogliamo rappresentare in sistema binario, l'uscita fornirà lo stesso numero ma in formato a 7 segmenti. Per rappresentare i numeri dall'1 al 9 - con cui lavoriamo in questo progetto - abbiamo bisogno di quattro bit se lavoriamo in binario e di 7 bit se lavoriamo in codice a 7 segmenti, quindi il chip dovrà avere 4 ingressi e almeno 7 uscite.

Tale soluzione è dedicata a questo esperimento, tuttavia il programma del PIC è predisposto per fare a meno del driver (ma questo corrisponde a un altro esperimento) e realizzare esso stesso le sue funzioni; in questo caso utilizzeremo la porta B come abbiamo visto nella spiegazione del progetto.

Collegamenti alla porta C di DG06.

Collegamenti del cavetto ai catodi dei LED.

Collegamenti fra i catodi dei LED e la scheda dei driver.

Dettaglio del collegamento alla scheda dei driver.

Il montaggio

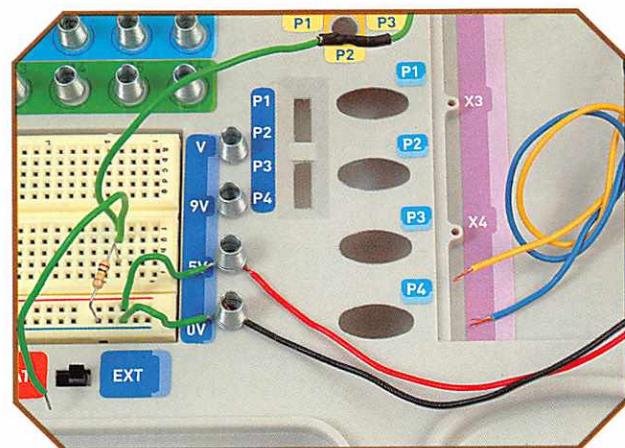
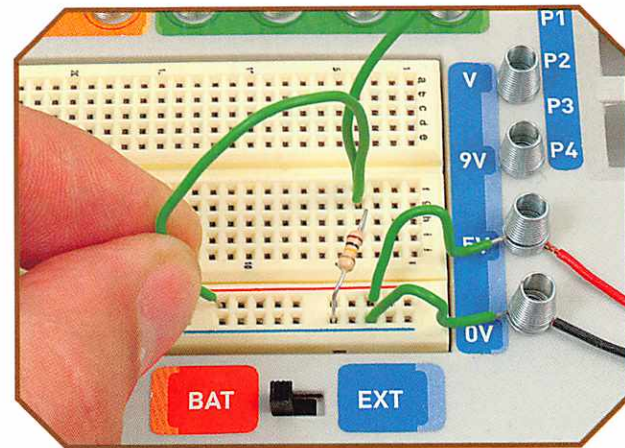
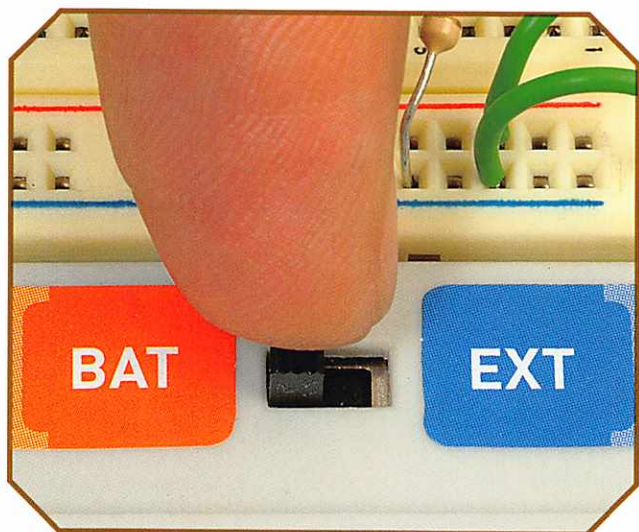
Se rispettiamo il montaggio che avevamo effettuato per il primo esperimento del circuito e colleghiamo mediante un cavetto l'altro estremo della matrice dei LED, cioè quella corrispondente ai catodi con il connettore J24 della scheda DG02, che contiene i driver a 7 segmenti nella Zona 1 del Laboratorio, raggiungeremo il nostro scopo.

Il cavetto che era già installato univa il connettore J64 (uscite RC0:RC3) della scheda DG06 agli anodi dei diodi LED, come possiamo vedere.

In altre parole, in questo caso i LED rimangono collegati in serie e non si illuminano, perché sono attraversati da una corrente molto bassa, sufficiente per attivare il driver, ma insufficiente per ottenere una corretta illuminazione degli stessi.

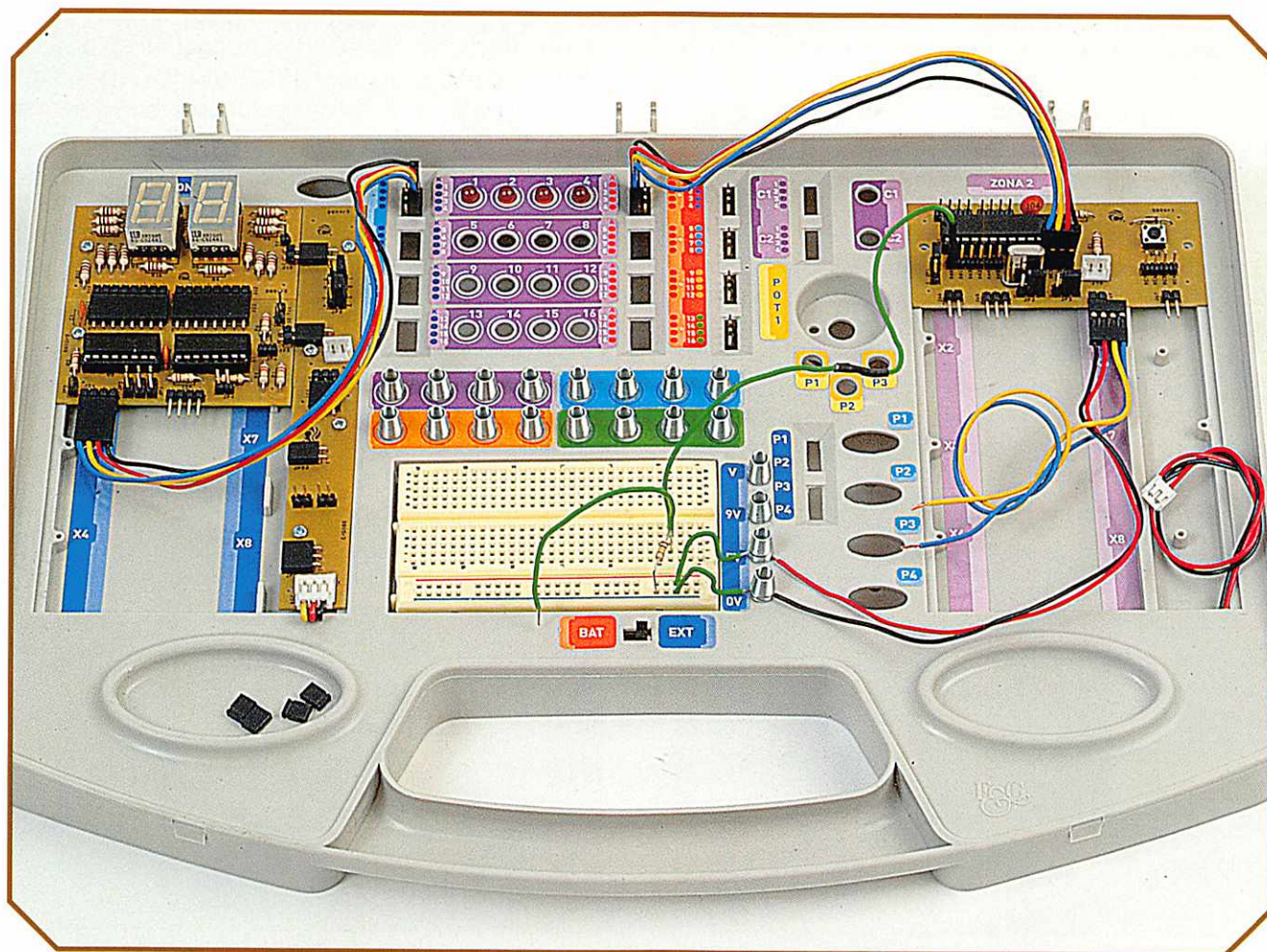
Il montaggio, seguendo le illustrazioni, risulterà rapido e semplice. I quattro terminali del connettore J24 sono collegati direttamente al driver del display 4511, quindi risulta indispensabile mantenere la corrispondenza con i colori dei fili, perché invertendo il collegamento del cavetto la sequenza risulterà al rovescio, e la visualizzazione non si produrrà o avverrà in modo errato. Non si deve utilizzare il connettore J25.

Ingressi

Dato che il funzionamento del PIC è lo stesso di quello dell'esperimento precedente, l'ingresso si implementa nello stesso modo, e si

Collegamenti dell'alimentazione e dell'ingresso di controllo.

Utilizzo dell'ingresso di controllo.

Prima di spostare su BAT bisogna verificare tutti i collegamenti.

utilizza il terminale 0 della porta RA e una resistenza da 100 K. Quando l'ingresso RA0 è a livello "1", si visualizzano i numeri in modo sequenziale e quando passa a livello "0", si vede il numero casuale ottenuto.

Alimentazione

L'alimentazione della scheda DG06 si realizza tramite le due molle 0 e 5 V. Si utilizza un cavetto a quattro fili, dei quali ne useremo solamente due, il rosso per il positivo, collegato a 5 V e il nero per il negativo, collegato a 0 V come GND. Le schede DG01 e DG02 si alimentano attraverso la DG04, e devono essere installati i ponticelli di collegamento fra i terminali 1-2 di JP1 e JP2 della DG04. Infine il commutatore di alimentazione deve essere sulla posizione BAT.

Vista generale del laboratorio con l'esperimento.

Programma per i primi esperimenti con il PIC

Anche se abbiamo già provato il funzionamento del PIC con il programma scritto, la procedura corretta è realizzare il progetto passo a passo, iniziando col definire ciò che vogliamo fare e stabilendo l'hardware e il software necessari, e infine realizzare la verifica pratica.

Primi esperimenti

Di seguito inizieremo un progetto che comprende diversi esperimenti e lo descriveremo in modo molto breve:

– Generazione di un numero casuale tra 0 e 9 con presentazione in binario.

– Generazione di un numero casuale tra 0 e 9 con presentazione su display a 7 segmenti tramite un driver BCD/7 segmenti.

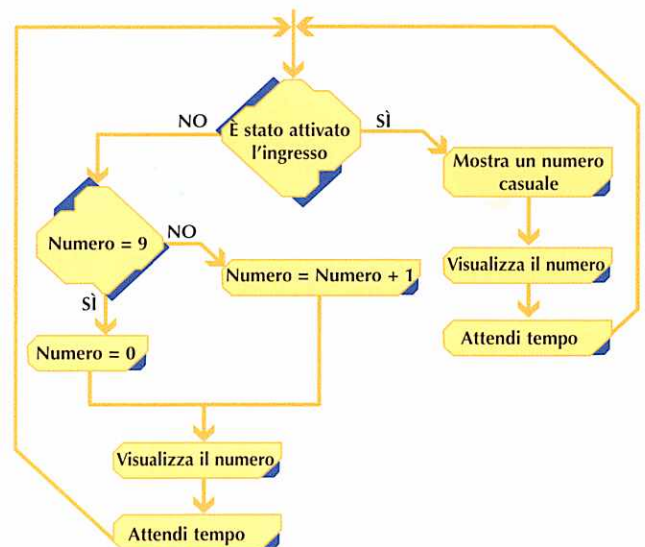
– Generazione di un numero casuale tra 0 e 9 con collegamento diretto al display senza utilizzare il driver precedentemente citato.

Anche se questo programma è già scritto nel PIC, lo spiegheremo nel dettaglio.

Analisi dell'enunciato

Si vuole generare un numero casuale tra 1 e 9 e visualizzarlo su quattro LED, che rappresentano il codice binario, e su un display a 7 segmenti.

Si dispone di un pulsante per attivare un ingresso del circuito che ferma il conteggio per visualizzare il numero casuale per tre secondi, tornando poi a ripetere, trascorso questo tempo, la visualizzazione sequenziale.

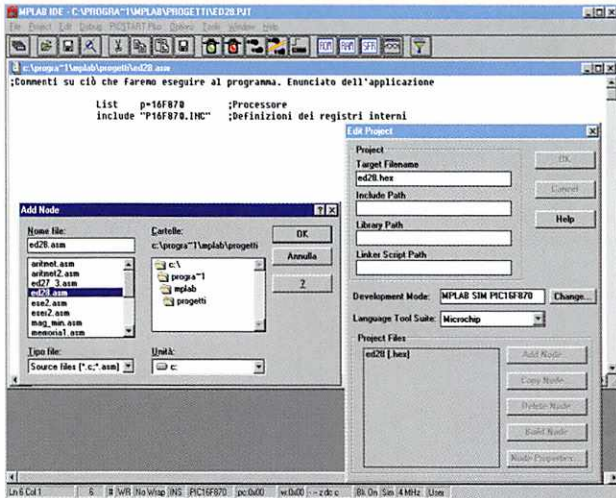
Per prima cosa bisogna capire in modo chiaro l'enunciato e portarlo in un organigramma.

Organigramma dell'applicazione.

L'organigramma non ha lo scopo di entrare nel dettaglio, ma ci aiuterà quando inizieremo a sviluppare il codice.

Fatto questo, il passo successivo consiste nell'elencare i dispositivi di cui abbiamo bisogno:

– Uscite: dato che il numero da rappresentare è compreso tra 1 e 9, con 4 bit, 4 uscite saranno sufficienti nel caso di attivare 4 LED o utilizzare un driver per convertire il codice binario a 7 segmenti. Tuttavia, se vogliamo attivare direttamente il display senza utilizzare i driver, oltre alle 4 uscite che vogliamo utilizzare per rappresentare il numero in binario, avremo bisogno di 8 uscite in più per ottenere direttamente dal PIC la conversione a 7 segmenti. Utilizzeremo 4 terminali della porta C (RC3:RC0) per ottenere il numero in binario, e 8 terminali della porta B (RB7:RB0) per ottenere il numero in codice a 7 segmenti.

	DIGIT	BINARIO	7 SEGMENTI b'Ogfedcba'
	0	0000	00111111
a	1	0001	00000110
	2	0010	01011011
f b	3	0011	01001111
g	4	0100	01100110
	5	0101	01101101
e c	6	0110	01111101
	7	0111	00000111
d	8	1000	01111111
	9	1001	01100111

Conversione di un digit a codice a 7 segmenti.

Creiamo un progetto,
lo editiamo e iniziamo a generare il codice.

```
;Commenti su ciò che fa' il programma. Enunciato dell'applicazione

                List    p=16F870            ;Processore
                include "P16F870.INC"       ;Definizioni dei registri interni

                org     0x00                ;Vector di Reset
                goto    Inizio
                org     0x05                ;Salva il vector di interrupt

Inizio          clrf    PORTB               ;Cancella i latch di uscita
                clrf    PORTC               ;Cancella i latch di uscita
                bsf     STATUS,RP0          ;Seleziona banco 1
                clrf    TRISB               ;Porta B si configura come uscita
                clrf    TRISC               ;Porta C si configura come uscita
                movlw   0x07
                movwf   ADCON1              ;Si configura la Porta A come I/O digitali
                movlw   0xFF
                movwf   TRISA               ;Porta A si configura come ingresso
                movlw   b'00000111'
                movwf   OPTION_REG          ;Prescaler da 256 per il TMR0
                bcf     STATUS,RP0          ;Seleziona il banco 0
```

Configuriamo i dispositivi del PIC,
in questo caso le porte di I/O.

```
Loop            clrwdt                      ;Aggiorna il WDT
                btfss   PORTA,0             ;RA0 è stato attivato?
                goto    Loop                ;Non ancora
                movf    TMR0,W              ;Adesso sì
                movwf   Numero              ;Acquisisce il valore del TMR0 (N° casuale)
```

È stato premuto l'ingresso?

```
Dividi:         movlw   d'9'
                subwf   Numero,W        ;Sottrae 9 al numero casuale
                movwf   Numero          ;Lo memorizza
                sublw   d'8'
                btfss   STATUS,C        ;Controlla se è minore di 8
                goto    Dividi          ;NO
                incf    Numero,F        ;Il numero è compreso fra 1 e 9
```

Adattamento del valore acquisito per ottenere
un numero fra 0 e 9.

– Ingressi: abbiamo bisogno solamente di un ingresso digitale, dato che in base a esso presenteremo il numero casuale o i numeri dall'1 al 9 in modo sequenziale. Configureremo RA0 della porta A come ingresso digitale.

– Dispositivi interni: dato che abbiamo due

```
Tabella:        addwf   PCL,F           ;Spostamento sulla tabella
                retlw   b'00111111'     ;Numero 0
                retlw   b'00000110'     ;Numero 1
                retlw   b'01011011'     ;Numero 2
                retlw   b'01001111'     ;Numero 3
                retlw   b'01100110'     ;Numero 4
                retlw   b'01101101'     ;Numero 5
                retlw   b'01111101'     ;Numero 6
                retlw   b'00000111'     ;Numero 7
                retlw   b'01111111'     ;Numero 8
                retlw   b'01100111'     ;Numero 9
```

Routine di conversione in codice a 7 segmenti.

```
Delay_20_ms:    bcf     INTCON,T0IF     ;Azzera il flag di overflow
                movlw   0xb1            ;Complemento hex. di 78
                movwf   TMR0            ;Carico il TMR0
Delay_20_ms_1   clrwdt                  ;Aggiorna il WDT
                btfss   INTCON,T0IF     ;Overflow del TMR0?
                goto    Delay_20_ms_1   ;Non ancora
                return
```

Routine di temporizzazione per eliminare i rimbalzi.

```
Delay_var:      bcf     INTCON,T0IF     ;Azzera il flag di overflow
                movlw   0x3c            ;Complemento hex. di 195
                movwf   TMR0            ;Carico il TMR0
Intervallo      clrwdt                  ;Aggiorna il WDT
                btfss   INTCON,T0IF     ;Overflow del TMR0?
                goto    Intervallo      ;Non ancora
                decfsz  Delay_cont,F    ;Decrementa il contatore di intervalli
                goto    Delay_var       ;Ripete l'intervallo di 50 ms
                return
```

Routine di temporizzazione di un tempo determinato multiplo di 50 ms in funzione di una variabile.

temporizzazioni, una per la rappresentazione del numero casuale e l'altra per visualizzare sequenzialmente i numeri, è necessario utilizzare i temporizzatori del PIC. Sarà sufficiente utilizzare il TMR0, anche se potremmo utilizzare un temporizzatore diverso a ogni temporizzazione.

Codice del programma

Possiamo iniziare a creare il codice che risolverà l'applicazione. A questo scopo apriremo MPLAB e creeremo un progetto, apriremo l'editor, daremo un nome al file in assembler e, mediante Edit Project, assegneremo questo file al progetto.

Di seguito presenteremo il nostro programma, ma voi potrete farne uno diverso che risolva comunque l'intendimento iniziale. Ogni progettista può presentare una soluzione differente, anche se, in linea generale, devono avere le stesse caratteristiche.

Configurazione

La prima cosa da fare quando iniziamo a programmare è configurare il nostro microcontroller, definendo il funzionamento delle porte di ingresso e uscita e dei dispositivi interni,

```
Dato:           movlw   d'9'
                movwf   Temporale       ;Inizializza il contatore del dato
RA0_1           clrwdt                  ;Aggiorna il WDT
                btfss   PORTA,0         ;Controlla se RA0 è a 1
                goto    Uscita          ;No, visualizza il numero casuale
                movf    Temporale,W     ;Numero da visualizzare
                movwf   PORTC           ;Visualizza sui LED
                call    Tabella         ;Conversione in BCD
                movwf   PORTB           ;Visualizza sul display
                movlw   d'1'
                movwf   Delay_Cont      ;Variabile di temporizzazione
                call    Delay_var       ;Temporizza 0,05"
                decfsz  Temporale,F     ;Numero successivo
                goto    RA0_1
                goto    Dato

                call    Delay_20_ms     ;Elimina rimbalzi

Uscita:         movf    Numero,W        ;Acquisisce il numero casuale
                movwf   PORTC           ;Porta il numero casuale in binario sui LED
                call    Tabella         ;Lo converte in 7 segmenti
                movwf   PORTB           ;Uscita sul display
                movlw   d'60'
                movwf   Delay_Cont      ;Inizializza variabile di temporizzazione
                call    Delay_var       ;Temporizza 3"
                clrf    PORTB           ;Scollega l'uscita
                clrf    PORTC           ;Scollega l'uscita sui LED

                goto    Loop

                end                     ;Fine del programma sorgente
```

Codice che risolve la visualizzazione.

se si lavora con essi. Il codice, quindi, dovrebbe avere la forma della figura.

Lettura dell'ingresso

Configurato il micro potremo già entrare nel programma in sé. Seguendo l'organigramma possiamo vedere che la prima cosa da fare è verificare la condizione dell'ingresso, per sapere se il pulsante è stato premuto oppure no. Se l'ingresso è attivo, RA0=1, inizieremo il programma, acquisendo il valore casuale che si otterrà dal WDT e scrivendo questo valore sulla variabile "Numero". Questa variabile dovrà essere definita all'inizio del programma. Nelle linee di codice dell'immagine si può vedere quanto spiegato in precedenza.

Adattamento del numero casuale

Il numero casuale è, mediante sottrazioni consecutive, diviso per 9. In questo modo l'ultimo resto sarà compreso tra 0 e 8, che verrà incrementato di una unità per ottenere alla fine un numero tra 1 e 9.

Subroutine

– Conversione. Per poter presentare i numeri direttamente sul display a 7 segmenti li dobbiamo convertire in questo codice (se si utilizza il driver per il display sarà sufficiente mandare il codice binario). A questo scopo creiamo una subroutine che verrà richiamata quando avremo bisogno di questo valore. Dato che questa subroutine contiene solamente una tabella di conversione, la chiameremo "Tabella".

```
                List    p=16F870        ;Processore
                include "P16F870.INC"   ;Definizioni dei registri interni

Numero          equ     0x20            ;Numero casuale
Delay_Cont      equ     0x21            ;Contatore di intervalli
Temporale       equ     0x22            ;Variabile temporale

                org     0x00            ;Vector di Reset
                goto    Inizio
                org     0x05            ;Salva il vector di interrupt
;------------------------------------------------------------------------
Tabella:        addwf   PCL,F           ;Spostamento sulla tabella
                retlw   b'00111111'     ;Numero 0
                retlw   b'00000110'     ;Numero 1
                retlw   b'01011011'     ;Numero 2
                retlw   b'01001111'     ;Numero 3
                retlw   b'01100110'     ;Numero 4
                retlw   b'01101101'     ;Numero 5
                retlw   b'01111101'     ;Numero 6
                retlw   b'00000111'     ;Numero 7
                retlw   b'01111111'     ;Numero 8
                retlw   b'01100111'     ;Numero 9
;------------------------------------------------------------------------
Delay_20_ms:    bcf     INTCON,T0IF     ;Azzera il flag di overflow
                movlw   0xb1            ;Complemento hex. di 78
                movwf   TMR0            ;Carico il TMR0
Delay_20_ms_1   clrwdt                  ;Aggiorna il WDT
                btfss   INTCON,T0IF     ;Overflow del TMR0?
                goto    Delay_20_ms_1   ;Non ancora
                return
;------------------------------------------------------------------------
Delay_var:      bcf     INTCON,T0IF     ;Azzera il flag di overflow
                movlw   0x3c            ;Complemento hex. di 195
                movwf   TMR0            ;Carico il TMR0
Intervallo      clrwdt                  ;Aggiorna il WDT
                btfss   INTCON,T0IF     ;Overflow del TMR0?
                goto    Intervallo      ;Non ancora
                decfsz  Delay_Cont,F    ;Decrementa il contatore di intervalli
                goto    Delay_var       ;Ripete l'intervallo di 50 ms
                return
;------------------------------------------------------------------------
Inizio          clrf    PORTB           ;Cancella i latch di uscita
                clrf    PORTC           ;Cancella i latch di uscita
                bsf     STATUS,RP0      ;Seleziona banco 1
                clrf    TRISB           ;Porta B si configura come uscita
                clrf    TRISC           ;Porta C si configura come uscita
                movlw   0x07
                movwf   ADCON1          ;Si configura la Porta A come I/O digitali
                movlw   0xFF
                movwf   TRISA           ;Porta A si configura come ingresso
```

Programma completo che risolve l'enunciato 1.

```
                movlw   b'00000111'
                movwf   OPTION_REG          ;Prescaler da 256 per il TMR0
                bcf     STATUS,RP0          ;Seleziona il banco 0

Loop            clrwdt                      ;Aggiorna il WDT
                btfss   PORTA,0             ;RA0 è stato attivato?
                goto    Loop                ;Non ancora
                movf    TMR0,W              ;Adesso sì
                movwf   Numero              ;Acquisisce il valore del TMR0 (N° casuale)

                call    Delay_20_ms         ;Elimina rimbalzi
        ;--------------------------------------------------------------------------
Dividi:         movlw   d'9'
                subwf   Numero,W            ;Sottrae 9 al numero casuale
                movwf   Numero              ;Lo memorizza
                sublw   d'8'
                btfss   STATUS,C            ;Controlla se è minore di 8
                goto    Dividi              ;NO
                incf    Numero,F            ;Il numero è compreso fra 1 e 9
        ;--------------------------------------------------------------------------
Dato:           movlw   d'9'
                movwf   Temporale           ;Inizializza il contatore del dato
RA0_1           clrwdt                      ;Aggiorna il WDT
                btfss   PORTA,0             ;Controlla se RA0 è a 1
                goto    Uscita              ;No, visualizza il numero casuale
                movf    Temporale,W         ;Numero da visualizzare
                movwf   PORTC               ;Visualizza sui LED
                call    Tabella             ;Conversione in BCD
                movwf   PORTB               ;Visualizza sul display
                movlw   d'1'
                movwf   Delay_Cont          ;Variabile di temporizzazione
                call    Delay_var           ;Temporizza 0,05"
                decfsz  Temporale,F         ;Numero successivo
                goto    RA0_1
                goto    Dato

                call    Delay_20_ms         ;Elimina rimbalzi

Uscita:         movf    Numero,W            ;Acquisisce il numero casuale
                movwf   PORTC               ;Porta il numero casuale in binario sui LED
                call    Tabella             ;Lo converte in 7 segmenti
                movwf   PORTB               ;Uscita sul display
                movlw   d'60'
                movwf   Delay_Cont          ;Inizializza variabile di temporizzazione
                call    Delay_var           ;Temporizza 3"
                clrf    PORTB               ;Scollega l'uscita
                clrf    PORTC               ;Scollega l'uscita sui LED
                goto    Loop

                end                         ;Fine del programma sorgente
```

Programma completo che risolve l'enunciato 2.

Il funzionamento di questa tabella è molto semplice, dato che si tratta di sommare al contatore di programma il valore che vogliamo tradurre, in questo modo il contatore di programma va sulla posizione dove si trova il valore tradotto e ritorna dalla subroutine.

– Temporizzazioni. Nel nostro esempio abbiamo creato due temporizzazioni. Una ha lo scopo di eliminare i rimbalzi o i valori errati che possono generare gli elementi elettromeccanici. Ha una durata fissa di 20 ms e questo valore deriva dal fatto che il PIC lavora a una frequenza di 4 MHz e il TMR0 evolve ogni microsecondo, quindi se vogliamo temporizzare 20.000_s (20 ms) con un predivisore di 256, il TMR0 dovrà contare 78 eventi (78x256). Il valore 78 equivale a 0x4e h e dato che il TMR0 è ascendente, dovremo caricare il suo complemento a 1 (0xb1 hex.).

Mediante una nuova routine di temporizzazione programmeremo il tempo da attendere nella rappresentazione sequenziale o nella visualizzazione del numero casuale. Il ragionamento per la nuova temporizzazione è simile al precedente: la velocità di lavoro è di 4 MHz , quindi il TMR0 si incrementa ogni microsecondo. In questo modo il TMR0 deve contare 195 eventi che, con un predivisore di 128, producono un intervallo totale di 50.000 µs - 50 ms (195x256). Il valore 195 deve essere espresso in esadecimale (c3) e, dato che TMR0 è ascendente, bisognerà caricare il suo complemento (3c h). Questo intervallo di 50 ms si ripete tante volte quante indicate dalla variabile "Delay_cont"; per questa ragione il ritardo (in inglese delay) minimo è di 50 ms ("Delay_cont=1) e quello massimo di 12,8" (Delay_cont=255). Nel nostro caso la temporizzazione di 3 secondi nella visualizzazione del numero casuale la realizzeremo con Delay_cont=60.

Abbiamo quindi una nuova variabile che deve essere dichiarata all'inizio del programma.

Con questi esempi si può verificare come sia semplice fare una temporizzazione.

Presentazione dei numeri

Manca poco per avere il nostro primo grande programma risolto, ma manca ancora da fare la parte che può essere considerata la più difficile, risolvere l'enunciato scrivendo il codice in assembler. Abbiamo fatto la configurazione dei dispositivi, l'acquisizione del numero casuale, creato le temporizzazioni, la conversione a 7 segmenti, ma ora dobbiamo assemblarlo in modo più pratico per risolvere l'enunciato iniziale.

Nelle illustrazioni si può vedere come è stato strutturato il programma in modo da risolvere in modo semplice il progetto. Il programma è debitamente commentato per far risultare più semplice la sua comprensione.

Potete verificare come sia stata creata una nuova variabile che servirà per contenere i valori in modo temporale e come vengono chiamate le subroutines o vengono eseguiti i salti alle altre parti del programma commentate in precedenza.